Semester Project

On the Effect of Co-location Information on Location Privacy

Alexandra-Mihaela Olteanu

Phd Student
EPFL
School of Computer and Communication Sciences

January 27th, 2014

Professor
Prof. Jean-Pierre Hubaux
EPFL - Lausanne, CH

Supervisor
Kévin Huguenin
EPFL - Lausanne, CH

Supervisor
Reza Shokri
ETH - Zurich, CH

1 Introduction

In past years there has been an increased modernization of mobile devices, most of which are GPS equiped. Consequently, an abundance of applications that collect location directly from these devices, in order to provide personalized and contextual services for users, emerged. People trade in their location in order to search for points of interest, such as restaurants, shops, landmarks, or directions from their phone. Location-based activities have become popular on online social networking sites (OSNs) as well. People can share and interact more with their friends when they share their location too. Today's modern mobile devices typically have cameras as well, which, alongisde high-speed mobile internet connections and the popularity of OSNs, resulted in astonishing and still increasing amount of photos uploaded online. More than 250 billion photos are uploaded on Facebook, with 4,000 photos being uploaded per second [4] and 1.54 million are uploaded on Flickr daily [2]. For an enriched experience, people can, and often do, disclose their location to their friends when sharing their pictures. Furthermore, most picture taking devices (phones, cameras) can embed metadata information (*e.g.,* EXIF headers) within a digital photograph, such as location information or user identity, and some OSNs do allow geo-tagged pictures to be uploaded.

Such trends have naturally raised concern regarding user privacy. As a response to these issues, various Location Privacy Protection Mechanisms (LPPMs) have been proposed. They include obfuscating a person's true location by hiding it, adding noise or reducing precision. There is extensive work in the literature including experimental and theoretical analysis of user location privacy under sporadic user location exposure in the presence of LPPMs. Shokri et al [5] provide a formal framework for the analysis of LPPMs, propose an optimal inference algorithm and a convincing metric to quantify location privacy based on adversary incorectness. They measures the privacy of mobile users for different LPPMs, showing that LPPMs can protect users' location privacy, but still leave room for harmful activities, when users don't use the right protection setttings.

Online privacy mostly gives users control over individual settings and content, not actions or content of others. Yet, other people can disclose or facilitate the inference of sensitive location information about their friends on OSNs. Such location information about more users, or *co-location*, can be declared on OSNs via status updates, declared or detectable from posted pictures that tag other people, contain metadata or via face recognition software. Co-location can also be infered based on Bluetooth sharing or from traffic requests with the same IP (users behind the same NAT). Srivatsa et al [6] show that de-anonymizing mobility traces is possible using a social graph as a side channel, which they infer from co-location information. Henne et al [1] discuss privacy implications of the emerging trend of geo-tagged photos by others. But, despite co-locations being so widely available, we are not aware of any work studying directly the impact of co-locations on user privacy. When combined with individual location information, co-location could improve the location inference. Take the example of two users, one declares he is with the other in some location - the other's location can be immediately infered. Or two users independently disclose their location and one of them later discloses the fact that they are together - without even knowing the exact location of the co-location, it is easier to infer the users' location at the time of co-location, based on their habits and physical constraints of mobility. Or the users individually disclose their obfuscated location at some time instant and one of them declares they are togehter - deobfuscating their individual location becomes easier with the knowledge that an obfuscated version of the same actual location is reported by both. While co-locations provide more information to an attacker, they add complexity, making infering location a joint optimization among all users. Without taking into account co-locations, the localization problem can be solved as an isolated optimization problem for each user. Co-locations add user dependencies which require a solution to be jointly evaluated. Taking into account all possible combinations for a solution, leads to an exponential explosion of the size of the possible solution space.

In this work, we formalize the localization problem by expanding the model presented in [5] to include co-locations. We analize and formally prove its complexity and propose two optimal inference algorithms. We evaluate one of our algorithms on a real dataset [8], [9], [7] and quantify the additional privacy loss due to co-locations on user privacy. We confirm the intuition that co-locations lead to privacy loss by providing more information to an attacker. We also look at how different individual LPPM settings impact privacy for users with co-locations, showing that an individual user has less control over his privacy through his individual privacy settings, when co-locations that include him can be observed.

2 Formalization

We assume a model with N users, with GPS-equipped devices that sporadically disclose their location and co-locations. Users move within a set of M geographic regions $R = \{r_1, ..., r_M\}$ and their location is observed at

time instants ranging from 1 to T. We consider an honest-but-curious adversary, typically the service provider, but could be anyone that can collect publicly available users' data from OSDNs (check-ins, pictures) or an ISP having access to users' traffic, from which co-locations can be inferred. Such an adversary can gather **knowledge** about users' mobility and incorporate it into a Markovian model. For a user u, we denote this model by P^u - an $M \times M$ matrix, where P^u_{ji} represents the probability that user u will move from region r_i to region r_j in one unit of time. The adversary can build such knowledge from users' past traces, as described in [5].

At any point in time $t \in \{1, ..., T\}$, the adversary can make either one or both types of **observations**:

- **A single obseration for** u, which we denote by $(\mathbf{u@_t r})$ (at time t user u is in region r). The observed region r can be an obfuscated version of the true region.
- **A social observation, or a co-location, for** u_i **and** u_j. This can be obtained from *either* u_i's *or* u_j's behavior. We denote such an observation by $(\mathbf{u_i \leftrightarrow_t u_j})$ meaning that at time t users u_i and u_j are together.

Notice that a typical co-location observation does not disclose the users' location, only the fact that they are together. A simple co-location cannot be hidden or obfuscated, but it is possible for it to not be declared by the users or missed by the attacker. In reality, a user can also declare the location and tag or mention friends that are with him, for example Alice can check-in on Facebook with Bob, at EPFL. We represent such information by a co-location observation ($\mathbf{Alice \leftrightarrow_t Bob}$) and a single observation ($\mathbf{Alice@_t EPFL}$), where, again, the declared location can be obfuscated. We can also include scenarios when more than two users are together at the same time by multiple pairwise co-location observations.

We assume any user u can have an LPPM, which we generically model to include location hiding and/or location obfuscation.

- **location hiding** - the true user's location is hidden with probability λ_u
- **location obfuscation** - if not hidden, an obfuscated version of the true location can be disclosed. Any function $f : R \times R \to \mathbb{R}$ s.t. $\forall x \in R, \sum_{y \in R} f_x(y) = 1$ can be used as pdf for the obfuscation mechanism.

These protection mechanisms are applied individually for each user, independent of the time and actual location at that time. The adversary observes a user's location as the output of the LPPM. Specifically, if the location at time t is r, the adversary can either not observe it at all (with probability λ_u), or observe *any* obfuscated r' for which $f_r(r') > 0$, each with probability $(1 - \lambda_u) \cdot f_r(r')$. Notice that due to the randomness of the processes, the same actual location r can be perceived as a different obfuscated location at different time instants.

As users move over time, they produce individual **actual traces**, while their location (reported or infered) is observed. These potentially obfuscated observed locations form **obfuscated traces**, which can be incomplete. The location at some time instant cannot be observed either due to location hiding, or technical issues on the user side (such as no internet/GPS connectivity). The adversary only observes these obfuscated traces of single observations. Over time, users also report co-locations (via posted pictures tagging each other, or via location disclosing status updates mentioning friends), which the adversary can observe.

We consider these single and co-location **observations** over the time interval [1 T] to be the input for the adversary. Together with his prior **knowledge** mobility model for each user, he aims to solve the **localization problem**, i.e. find the backwards mapping from observed location at some time instant t for some u to the actual location of u at t.

Notations

- $P^U = \{P^{u_1}, ..., P^{u_N}\}$
- π^{u_i} - the steady state vector of $P^{u_i}, i.e.\ \pi^{u_i} \in R^M s.t.\ \pi^{u_i} \cdot P^{u_i} = \pi^{u_i};\ \pi^U = \{\pi^{u_1}, ..., \pi^{u_N}\}$
- $a_{u_i}(t)$ – actual location of u at t, $\underline{a}(t) = \{a_{u_1}(t), ..., a_{u_N}(t)\}$
- $\theta_{u_i}(t)$ - observed location of u_i at t, $\underline{\theta}(t) = \{\theta_{u_1}(t), ..., \theta_{u_N}(t)\}$
- $@_{t_1..t_2}$, $\leftrightarrow_{t_1..t_2}$ - the set of all single, co-location observations for all users and time instants in the interval [t_1 t_2], respectively.
- $u_i\ @_{t_1..t_2}$ - the set of all single observations for u_i at time instants in the interval [t_1 t_2], respectively.
- $\underline{v} \in S^d | \underline{v}_i = x$ - any d-dimentional vector with elements in S with the value at position i fixed to x.

2.1 A Forward-Backward Algorithm with joint variables

Without co-locations, the localization problem translates to solving a typical HMM model for which a forward-backward algorithm is a known solution. However, with co-locations, users locations are no longer independent and incorporating this dependancy is not entirely trivial. In our first approach, we consider user's locations at every instant jointly, and propose a forward-backward algorithm.

Notations

- $\forall \underline{r}, \underline{\rho} \in R^N : P^U_{\underline{\rho}\underline{r}} = \prod_{i=1,N} P^{u_i}_{\rho_i r_i},\ \pi^U(\underline{r}) = \prod_{i=1,N} \pi^{u_i}(r_i)$
- $\forall \underline{r} \in R^N : f_{\underline{r}}(\underline{\theta}(t)) = \prod_{i=1,N s.t. \exists\ u_i\ @_t} f_{r_i}(\theta_{u_i}(t))$
- $\mathbf{1}(\underline{r}, \leftrightarrow_t) = \wedge_{u_i \leftrightarrow_t u_j \in \leftrightarrow_t}(r_i = r_j)$. This is equal to 1 for a joint region space that is compatible with co-location observations at t and 0 otherwise. For example, with $N = 2, M = 2, R = \{0, 1\}$, if the two users that have a colocation at t, $\mathbf{1}([0, 0], \leftrightarrow_t) = \mathbf{1}([1, 1], \leftrightarrow_t) = 1$ and $\mathbf{1}([0, 1], \leftrightarrow_t) = \mathbf{1}([1, 0], \leftrightarrow_t) = 0$ (there can't be a co-location at t if users are not in the same region). If the users do not have a co-location at t', $\mathbf{1}([r_1, r_2], \leftrightarrow_{t'}) = 1, \forall r_1, r_2 \in \{0, 1\}$.

For any set U of N users, $t \in [0\ T]$ and $\underline{r} \in R^N$ we define:

$$\alpha^U_t(\underline{r}) \coloneqq \Pr(\ @_{1..t}\ , \leftrightarrow_{1..t},\ \underline{a}(t) = \underline{r} \mid P^U) \tag{2.1}$$

Theorem 1. *For any $\underline{r} \in R^N$ and any $t \in [0..T]$, $\alpha^U_t(\underline{r})$ can be computed by the following formula:*

$$\alpha^U_t(\underline{r}) = \begin{cases} \pi^U(\underline{r}) & \text{if } t = 0 \\ \mathbf{1}(\underline{r}, \leftrightarrow_t) \cdot f_{\underline{r}}(\underline{\theta}(t)) \cdot \sum_{\underline{\rho} \in R^N} P^U_{\underline{r}\underline{\rho}} \cdot \alpha^U_{t-1}(\underline{\rho}) & \text{if } t > 0 \end{cases} \tag{2.2}$$

For any set U of N users, $t \in [0\ T]$ and $\underline{r} \in R^N$ we define:

$$\beta^U_t(\underline{r}) \coloneqq \Pr(\ @_{t+1..T}\ , \leftrightarrow_{t+1..T} \mid \underline{a}(t) = \underline{r},\ P^U) \tag{2.3}$$

Theorem 2. *For any $\underline{r} \in R^N$, any $t \in [0..T]$, $\beta^U_t(\underline{r})$ can be computed by the following formula:*

$$\beta^U_t(\underline{r}) = \begin{cases} 1 & \text{if } t = T \\ \sum_{\underline{\rho} \in R^N} \mathbf{1}(\underline{\rho}, \leftrightarrow_{t+1}) \cdot \beta^U_{t+1}(\underline{\rho}) \cdot f_{\underline{\rho}}(\underline{\theta}(t+1)) \cdot P^U_{\underline{\rho}\underline{r}} & \text{if } t < T \end{cases} \tag{2.4}$$

Notice there are no single observations at $t = 0$, but we define our variables at 0 in order to facilitate their iterative computation.

Theorem 3. *For any user u_i and any t, the following holds:*

$$\forall r \in R : \Pr(a_{u_i}(\mathbf{t}) = \mathbf{r} |\ @_{1..T}\ , \leftrightarrow_{1..T}, P^U) = \frac{\sum_{\underline{r} \in R^N | \underline{r}_{\mathbf{u}} = \mathbf{r}} \alpha^U_t(\underline{r}) \cdot \beta^U_t(\underline{r})}{\sum_{\underline{r'} \in R^N} \alpha^U_t(\underline{r'}) \cdot \beta^U_t(\underline{r'})} \tag{2.5}$$

Complete derivations for 2.2, 2.4 and 2.5 can be found in our technical report [3].

Formula 2.5 can be used to perform a localization attack for any user, at any time instant. Such an attack can be done in two stages. The first stage includes the computation of α and β variables with a time complexity of $\mathcal{O}(T \cdot M^{2N})$. In the second stage, computing the probability distributions for *all* users at *all* time instants can be done in $\mathcal{O}(N \cdot T \cdot M^N)$. This result, which shows the algorithm's complexity is exponential in the number of users, proves the complexity of the problem.

2.2 A dynamic programing approch with individual variables

We propose another solution to the localization attack. For any user $u_i \in U$, for any two time instants $t_b, t_e \in [1...T], t_b \leq t_e$ and for any regions $r_b, r_e \in R$, we define:

$$\gamma^i_{t_b,t_e}(r_b, r_e) \coloneqq \Pr(u_i\ @_{t_b+1..t_e}\ , a_{u_i}(t_e) = r_e | a_{u_i}(t_b) = r_b, P^{u_i}) \tag{2.6}$$

Theorem 4. *For any $u_i \in U$, any regions $r_b, r_e \in R$ and any $t_b, t_e \in [0..T]$, $\gamma^i_{t_b,t_e}(r_b, r_e)$ can be computed by the following formula:*

$$\gamma^i_{t_b,t_e}(r_b, r_e) = \begin{cases} 0 & \text{if } t_b = t_e > 0 \text{ and } r_b \neq r_e \\ 1 & \text{if } t_b = t_e > 0 \text{ and } r_b = r_e \\ f_{r_e}(\theta_{u_i}(t_e)) \cdot \pi^{u_i}(r_e) & \text{if } t_b = 0, t_e = 1 \\ f_{r_e}(\theta_{u_i}(t_e)) \cdot \sum_{\rho \in R} (P^{u_i}_{r_e\rho} \cdot \gamma^i_{t_b,(t_e-1)}(r_b, \rho)) & \text{if } t_e > t_b \geq 0 \end{cases} \tag{2.7}$$

Notations

- C - the total number of co-location observations between any two users
- C^i - the total number of co-location observations user u_i is a part of; $\sum_{i=1,N} C^i = 2C$
- $\underline{k}^i$ - the time instants for which u_i is part of a co-location observation
- $\underline{\rho} \in R^{C+1}$ for some time instant t - a vector representing a possible combination of regions, each corresponding to a time instant in $\{t\} \cup \bigcup_{j=1,N} \underline{k}^j$; $\forall \underline{t}' \in \mathbb{N}^d, d < C, \underline{\rho}[\underline{t}']$ - the subset of values in $\underline{\rho}$, corresponding to time instants in $\underline{t}'$.
- For any user u_i, for some dimension $d < C$, any time instants $\underline{t} \in \mathbb{N}^d$, any regions $\underline{\rho} \in R^d$, each corresponding to a time instant in $\underline{t}$:

$$\underline{\gamma}^i(\underline{\rho}) \coloneqq \sum_{r \in R} \gamma^i_{0,t_1}(r, \rho(t_1)) \cdot \left(\prod_{j=1,d} \gamma^i_{t_j,t_{j+1}}(\rho(t_j), \rho(t_{j+1})) \right) \cdot \sum_{r \in R} \gamma^i_{t_d,T}(\rho(t_d), r))$$

Theorem 5. *For any user u_i and any t, the following holds:*

$$\forall r \in R : \Pr(a_{u_i}(\mathbf{t}) = \mathbf{r} |\ @_{1..T}, \leftrightarrow_{1..T}, P^U) = \frac{\sum_{\underline{\rho} \in R^{C+1} | \underline{\rho}_{\mathbf{t}} = \mathbf{r}} \left(\underline{\gamma}^i(\underline{\rho}[\underline{k}^i \cup \{\mathbf{t}\}]) \cdot \prod_{\substack{j=1,N \\ j \neq i}} \underline{\gamma}^j(\underline{\rho}[\underline{k}^j]) \right)}{\sum_{\underline{\rho}' \in R^{C+1}} \left(\underline{\gamma}^i(\underline{\rho}'[\underline{k}^i \cup \{\mathbf{t}\}]) \cdot \prod_{\substack{j=1,N \\ j \neq i}} \underline{\gamma}^j(\underline{\rho}'[\underline{k}^j]) \right)} \tag{2.8}$$

Complete derivations for 2.7 and 2.8 can be found in the technical report [3].

Formula 2.8 can be used to perform a two-stage localization attack for any user, at any time instant. The first stage includes the computation of the γ variables, with time complexity $\mathcal{O}(N \cdot M^2 \cdot T^2)$. The second stage consists of computing the probability distributions for *all* users at *all* time instants, with a time complexity of $\mathcal{O}(N \cdot T \cdot M^{C+1})$. The complexity of the problem is expressed in an exponential with respect to the total number of co-location observations, C, which can be at most $(N/2+1)T$ (for any time instant, $N/2+1$ observations are enough to represent the extreme case of a co-location for all N users). The choice of algorithm to employ for an optimal localization attack could then be made depending on the dataset. If the dataset contains few co-locations, the γ-attack would be a better choice, while with many co-locations (more than N) the $\alpha\beta$-attack would be faster.

2.3 Quantifying user privacy

The localization attack can be implemented using either formula 2.5 or 2.8, for any user and any time instant, including those for which no obfuscated single observation exists. In other words, the attacker received sparse obfuscated traces, but can infer location at any time instant.

As shown in [5], a good measure for user privacy is adversary incorectness. We quantify user privacy by the (normalized) **expected adversary error**. For a given t and u, whose true actual location at t is r^* and $\forall r \in R, \Pr(a_u(t) = r)$ - the output of the localization attack for u at t, we measure u's privacy at t by:

$$LP(u,t) = \sum_{r \in R} \Pr(a_u(t) = r) \cdot d(r, r^*);\ LPN(u,t) = \frac{LP(u,t)}{\max_{r', r'' \in R} d(r', r'')} \tag{2.9}$$

where $d(r_1, r_2)$ is a distance function defined on $R \times R \to \mathbb{R}$.

For a set of N users U, we compute the average user privacy across T time instants, having the same unit of measure as the region distance function, and its normalized version, taking values in [0 1], by:

$$LP(U) = \frac{\sum_{u \in U, t \in [1..T]} LP(u,t)}{N \cdot T};\ LPN(U) = \frac{\sum_{u \in U, t \in [1..T]} LPN(u,t)}{N \cdot T} \tag{2.10}$$

3 Evaluation

We implement the $\alpha\beta$-attack algorithm and evaluate it for $N = 2$. We use the GPS trajectory dataset, collected by Microsoft Research Asia [8], [9] and [7], which includes mobility traces for 182 users, mostly around Beijing,

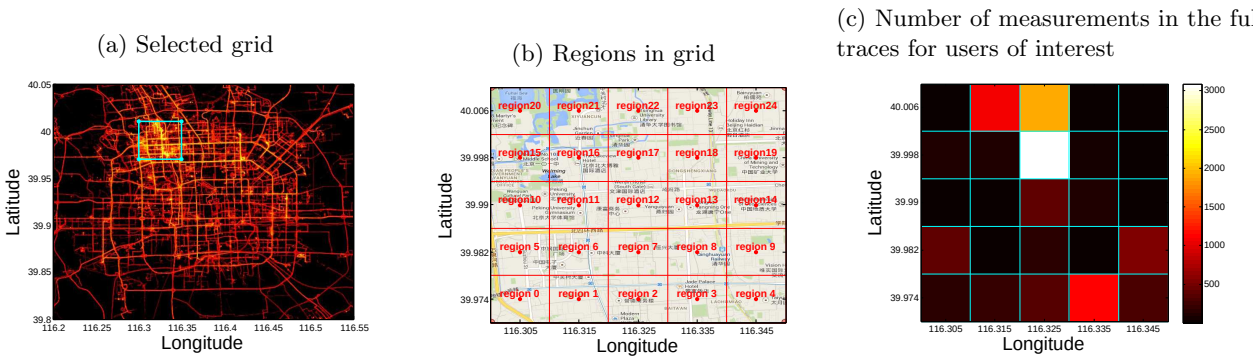


Figure 3.1: Evaluation dataset

China. We limit the observations to those with *longitude* $\in [116.3, 116.35]$ and *latitude* $\in [39.97, 40.01]$. The resulting area corresponds to a region with highly concentated data, as shown in Figure 3.1a.

We use the equi-rectangular projection to map the spherical coordinates (longitude, latitude) of all measurements within this area onto a 2D grid, in which the Euclidean distance between two projections is a good approximation of the Haversine Distance (*i.e.,* great-circle distance) between the original projected points. We use space discretization by splitting the selected grid into 25 approximately square grid cells (852m by 890m), each representing a region, as shown in Figure 3.1b. Given a user's original measurement in longitude and latitude, we consider the corresponding single observation to be the region cell in which the point falls.

Measurements in the dataset are given at heterogenous time intervals, ranging from 1-2 seconds to days or weeks. We down-sample the dataset in time by only keeping measuremnts at interval distance of 1 hour. When a measurement does not exist, we approximate it by the nearest measurement 1 hour before or after the time of interest.

For any two such processed user traces, we consider there to be a co-location between the two users at time instant t if the cells corresponding to position t are the same in both their traces. We find that only 961 of the possible $181 \cdot 182/2$ user pairs have at least one co-location in their traces. Of these 961 pairs, only 8 pairs have more than 100 co-locations: users 0 and 4 with 315 co-locations, users 0 and 30 with 280 co-locations, users 3 and 4 with 578 co-locations, users 3 and 30 with 407 co-locations, users 4 and 30 with 444 co-locations, users 126 and 167 with 553 co-locations, users 126 and 167 with 315 co-locations and users 153 and 163 with 123 co-locations. We select user pairs (3,4), (0,4) and (4,30) in our experiments (including results for the other user pairs with at least 100 co-locations is in progress).

Figure 3.1c shows the total number of available measurements for users in these pairs. We use their full traces to build user profiles (recall P^u_{ji} - the probability to move from region r_i to r_j in one time instant) by counting the number of occurences of r_i followed by r_j at consecutive time instants in the trace of user u. We perform the attack on multiple, *shorter*, random portions of the full trace, which we call the *actual trace of* u. Specifically, we set a desired number of co-location observations between two actual traces, which we denote by C, and obtain pairs of actual traces such that exactly C co-locations can be observed in the actual traces.

For all actual traces, we apply location hiding and/or obfuscation. The result represents the attacker's input of obfuscated single observations. We implement location obfuscation by reporting a random location in a neighbourhood of the real location, each with equal probability. We define the neighbourhood of any region r, denoted by $N(r, kf_u)$, as the set of all regions r' that satisfy: $d_{Mh}(r, r') \leq kf_u$, where $d_{Mh}(r, r') \coloneqq |r_x - r'_x| + |r_y - r'_y|$ denotes the Manhattan distance in the 2D region grid, for a given kf_u. Hence, our location protection function, incorporating both location hiding and obfuscation, can be summarized by the below formula, where $r_\perp$ is a dummy location that symbolises a hidden location:

$$\forall r_a \in R, \forall r_o \in R \cup \{r_\perp\} : f_{r_a}(r_o) \coloneqq \begin{cases} \lambda_u & if\ r_o = r_\perp \\ \frac{(1-\lambda_u)}{|(N(r_a, kf_u))|} & if\ r_o \in N(r_a, kf_u) \\ 0 & otherwise \end{cases} \tag{3.1}$$

For example, for $\lambda = 0.4$ and $kf = 1$:
$f_4(r_\perp) = 0.4$; $f_4(3) = 0.2$; $f_4(4) = 0.2$; $f_4(9) = 0.2$ and all other $f_4(*) = 0$
$f_7(r_\perp) = 0.4$; $f_7(2) = 0.12$; $f_7(6) = 0.12$; $f_7(7) = 0.12$; $f_7(8) = 0.12$; $f_7(12) = 0.12$ and all other $f_7(*) = 0$

We aim to measure the influence on an attacker's corectness when using co-location observations, in addition

to single observations. To this end, we define ω, the proportion - expressed in % - of the existing C co-locations for a user pair that an attacker observes. An existing co-location could be missed by the adversary if neither of the users declare it or it cannot be inferred. For a given ω, we assume the adversary only observes $\omega\% \cdot C$ of C random co-locations.

For quantifying user privacy we use the metrics described in 2.9 and 2.10, with the Euclidean distance in the 2D grid as the distance function between two regions to obtain an average user privacy, $LP(U)$, measured in kilometers.

3.1 Measuring the gradual effect of co-locations

For simplicity, we firstly assume both users in a pair use the same LPPM settings, same λ and same kf. We vary λ in $[0, 0.1, ..., 0.9, 1]$, where 0 corresponds to location hiding not being used, and kf in $[0, 1]$, where 0 corresponds to obfuscation not being used. We select pairs of user actual traces, each having $C = 24$ co-locations and an average $T = 178$ and vary ω in $[0, 25, 50, 75, 100]$, where 0 corresponds to an attack using only single observations. U is the set of all users in the pairs, $\{0, 3, 4, 30\}$. We measure $LP(U)$ and $LPN(U)$ for all combinations of λ, kf and ω. For the same LPPM settings (λ, kf), the user privacy can vary due to the users selected, due to the actual traces selected for each user or due to the available co-locations used. In order to isolate the impact of co-locations, we vary the user pairs and, for each user pair, we consider 10 different actual traces. Additionaly, for each pair of actual traces, we consider 20 different obfuscation possibilities and 20 random ways to select $\omega\% \cdot C$ co-locations for the attacker to observe. Finally, we report as user privacy the median of $LP(U)$ and $LPN(U)$ over all the iterations for the same (λ, kf, ω).

In order to emphasize the difference in privacy which results from using co-locations, we define the relative **privacy loss** w.r.t the case when only single observations are used, measured in %, for different values of ω:

$$\Delta LP^{\omega}(U) = \frac{LP^{0}(U) - LP^{\omega}(U)}{LP^{0}(U)} \tag{3.2}$$

3.2 Measuring the impact of a friend's privacy settings with the use of co-locations

In another experiment, we assume users in a pair use different LPPM settings: $\lambda_{u_1}, \lambda_{u_2} \in [0, 0.2, 0.4, 0.6, 0.8, 1]$ and $kf_1, kf_2 \in [0, 1]$. We are interested in the impact on a user's privacy of co-locations together with the privacy settings of a user's friend. Without loss of generality, we call the first user in a pair *target* and the other *friend*. We select pairs of user actual traces, each having $C = 24$ co-locations and an average $T = 166$ and use all available co-locations for an attack. We measure $LP(U)$ and $LPN(U)$ for all combinations of λ_1, λ_2, kf_1 and kf_2, where U is the set of all *target* users in the pairs, $\{0, 3, 4, 30\}$ - here we perform the attack for both ordered versions of a pair ((u_1, u_2) and (u_2, u_1)). For each user pair, we generate 10 random pairs of actual traces and, for each of these, 20 random obfuscated traces.

For every combination $(kf_{\text{friend}}, \lambda_{\text{friend}})$, we define and measure the relative **privacy loss** for *target* users w.r.t. the case when only single observations are used, measured in %:

$$\Delta LP(U) = \frac{LP^{0}(U) - LP^{kf_{\text{friend}}, \lambda_{\text{friend}}}(U)}{LP^{0}(U)} \tag{3.3}$$

3.3 Results

In section 2.1 we proved the complexity of performing an attack which includes co-location observations along side single observations. Now we look at the improvement in the attacker's incorectness, or loss of user privacy, resulting from the use of co-locations. Figure 3.2 shows the results from our experiment described in 3.1. We notice that using co-locations results in a decrease of user privacy both with or without the use of location obfuscation. The only case when co-locations don't make a difference is when neither location obfuscation nor location hiding are used, a case for which user privacy is already down to 0 because location can be efficiently infered from fully disclosed actual traces based on the prior mobility knowledge alone. With location obfuscation, up to 28% more privacy can be lost, while in the case when obfucation is not used, the aditional privacy loss can be as high as 38%. Moreover, for the same LPPM setting λ, more privacy is lost as more co-locations are observed by the attacker (ω increases). We can conclude that *privacy is a decreasing function in the number of co-locations used in the attack*. This is in line with our original intuition that more information can decrease user privacy.

Secondly, we look at the privacy loss for a *target* user, under different privacy protection settings of a *friend*. The previous results show the case of a *friend* who is equally cautious as *target*, in terms of choosing his privacy protection settings. Figure 4.2 shows results from the experiment described in 3.2. Comparing Figure 3.2a with

Figure 4.2c and Figure 3.2b with Figure 4.2b, we observe that a $\lambda_{friend} < \lambda_{target}$(a *friend* who is less cautious than *target*) causes even more privacy loss for *target* than a similarly protected *friend*.

In fact, even if *friend* is more cautious than *target* ($\lambda_{friend} > \lambda_{target}$), the use of co-locations still results in a decrease of privacy for *target* w.r.t. the case when no co-locations are used. These observations are supported also by Figure 4.2a and Figure 4.2d. We can conclude that *the weaker protected a friend is* - either by hiding his location less often than *target*, or by not using location obfuscation - *the stronger the co-locations due privacy loss is for target*.

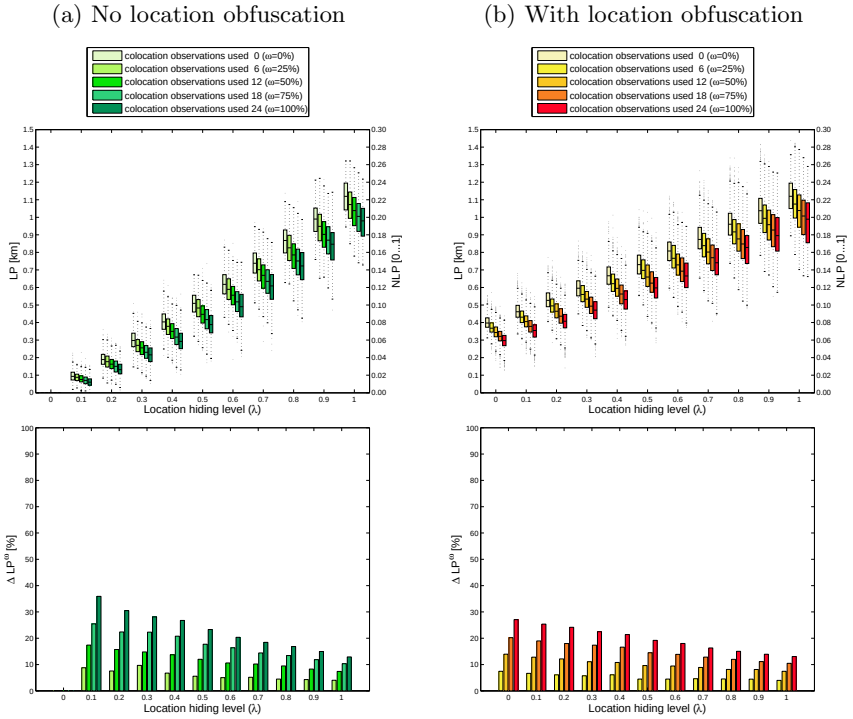
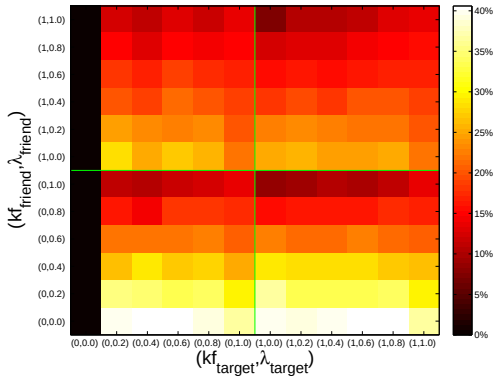


Figure 3.2: Gradual effect of co-locations over user privacy. Experiment setup as described in 3.1

Figure 4.1 shows *target*'s relative privacy loss (ΔLP), for all possible combinations of privacy settings for him and *friend*. We observe that the most dangerous cases are those when the friend rarely uses location hiding (low values for λ_{friend}). In this case, the relative privacy loss for *target* is very high - between 25-40% - **regardless** of his own privacy protection settings. This trully shows the power co-locations can give an attacker. A user can be very conscious about privacy threats and use conservative settings (hide and obfuscate his location), yet his privacy is highly threatened when co-locations with a poorly protected friend can be observed. The intuition behind this is that observing co-locations creates a connection between users: privacy is no longer soley determined by the user itself via the privacy settings he chooses for his LPPM, but it's a combination of the privacy settings of all users connected via co-locations. This implies that *a user has less control over his privacy, when co-locations can be declared by his friends, or infered.*

4 Conclusions and Future Work

Figure 4.1: A friend's effect on user privacy, measured by $\Delta LP(target)$.
Experiment setup as described in 3.2.
$kf_{target} \in [0,1], kf_{friend} \in [0,1]$



We analyze and formalize the localization problem from a social aspect, including information obtained from user co-location into an inference model. We prove its complexity and propose algorithms for a localization attack including both single and co-location observations. We implement and evaluate one of our algorithms for two users on a real dataset, showing that the additional information of co-locations leads to substantially more privacy loss and, in fact, this loss is an increasing function of the amount of co-locations available for the attack. We emphasize the influence of a friend's privacy settings on a user and show that poor privacy protection for a friend with which a user shares co-locations leads to more privacy loss for the user, despite his own strong privacy protection settings. This shows that, as the localization problem becomes a joint problem when using co-locations, individual privacy becomes a joint function of the users' privacy settings.

In the future, we want to measure the social observations impact for a tracking attack and test our optimal algorithms on more datasets, with more users, for a better understanding of co-

locations' impact on privacy. We aim to expand our model by including a probabilistic model for prior knowledge at the social level (such as $\Pr(u_i \ @_t \ r | u_j \ @_t \ r')$). The reason for this is that people's social relations can induce correlation between their locations at a particular point in time - *e.g.,* colleagues are likely to be together at lunch time, husbands and wives are likely to be together in the evenings.

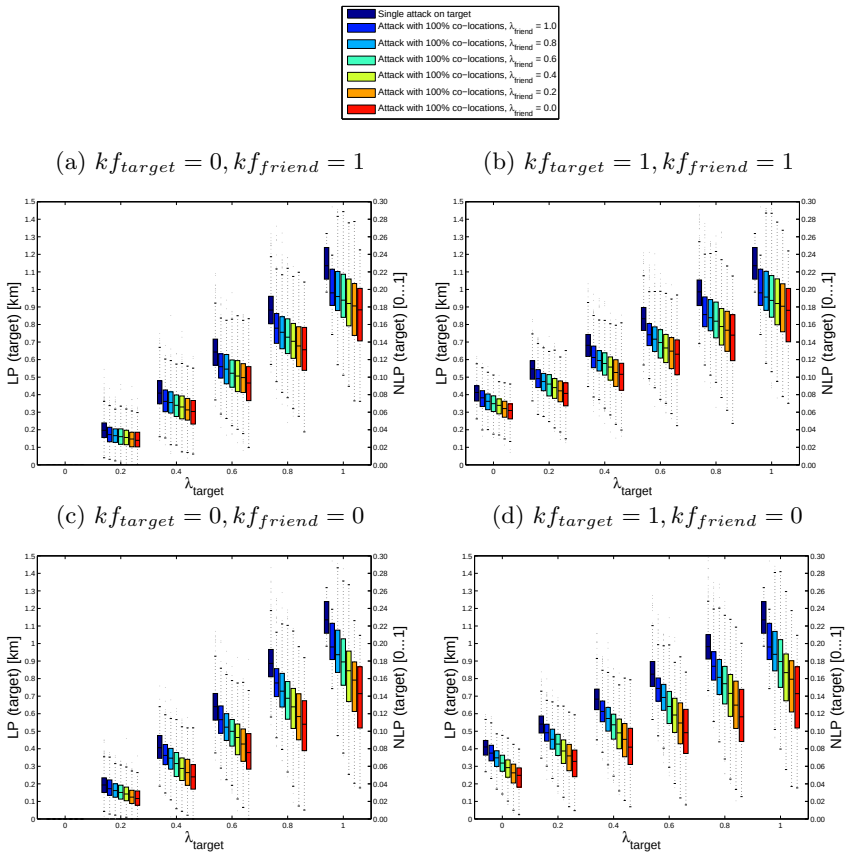


Figure 4.2: A friend's effect on user privacy. Experiment setup as described in 3.2

As some LPPMs also support user anonymization as part of privacy protection, *i.e.,* each user identifier is mapped to a user pseudonym, we plan to include this in our model. This adds another level of complexity to the localization problem, as user traces would be both obfuscated and anonymized. Finally, we want to reduce complexity through heuristics and compare these with our optimal algorithms. Examples of heuristics include bounding the time window for which co-locations are considered and bounding the number of friends (with co-locations) considered when performing an attack by using a graph partition that maximize the co-locations kept. The intuition here is that not all co-locations provide the same amount of information for an attack on u at t. A co-location at some t' far enough from t will induce a lower correlation between the actual location at t and that at t' than a co-location close to t will. And a direct co-location for u with a friend has a stronger influence in the attack on u than a co-location between a friend of a friend of u with someone u doesn't have direct co-locations.

Bibliography

[1] C. Szongott B. Henne and M. Smith. Snapme if you can: privacy threats of other peoples' geo-tagged media and what we can do about it. In *Proceedings of the Sixth ACM Conference on Security and Privacy in Wireless and Mobile Networks*, pages 95–106, April 2013.

[2] S. Schockaert O. V. Laere and B. Dhoedt. Finding locations of flickr resources using language models and similarity search. In *Proceedings of the 1st ACM International Conference on Multimedia Retrieval*, pages 48:1–48:8, October 2011.

[3] Alexandra Olteanu. On the effect of social information to location privacy - technical report.

[4] R. van Renesse W. Lloyd-S. Kumar Q. Huang, K. Birman and H. C. Li. An analysis of facebook photo caching. In *Proceedings of the Twenty-Fourth ACM Symposium on Operating Systems Principles*, pages 167–181, November 2013.

[5] R. Shokri, G. Theodorakopoulos, J.-Y. Le Boudec, and J-P Hubaux. Quantifying location privacy. In *Proc. of IEEE Symposium on Security and Privacy (S&P)*, pages 247–262, 2011.

[6] Hicks-M. Srivatsa, M. Deanonymizing mobility traces: Using social network as a side-channel. In *2012 ACM conference on Computer and communications security*, pages 628–637, 2012.

[7] Wei-Ying Ma Yu Zheng, Xing Xie. Geolife: A collaborative social networking service among user, location and trajectory. In *IEEE Data Engineering Bulletin*, pages 32–40, 2010.

[8] Xing Xie Wei-Ying Ma Yu Zheng, Lizhu Zhang. Mining interesting locations and travel sequences from gps trajectories. In *Proceedings of International conference on World Wild Web*, pages 791–800, 2009.

[9] Yukun Chen Xing Xie Wei-Ying Ma Yu Zheng, Quannan Li. Understanding mobility based on gps data. In *Proceedings of ACM conference on Ubiquitous Computing*, pages 312–321, 2008.